(17)

Reception System Large Stage Support

20 October 1988 Michael L. Gordon

1. Introduction

This paper will address the topic of support for "large" stages, those larger than are nominally distributed.

2. Background

Large stages may be beneficial in environments that can support the disk and memory resources necessary for their existence. Potentially, more objects can be locally available for use.

There is concern that if the universe of stage candidate objects is increased to exploit the increased storage capacity of the large stages, there may be negative consequences to performance of the regular sized stage, such as thrashing.

3. Plan

It is possible to mark objects for inclusion only in the large stage and be downward compatible with the existing base.

Of the two reserved storage candidacy values available in the object header, one of them will be assigned the new value: large stage no version check candidacy. Such an object will only be installed in systems with large stages. Of course, such a new feature requires enhancement to the stage file system. However, existing systems will handle such an object by treating it as a memory cache candidate and not installing it in the stage. The single deviation from the behavior of cacheable objects is that the object will not be written to the disk cache if its memory buffer is to be forfeited.

The reason that the candidacy includes the attribute no version check is that with only two reserved storage candidacy values available, it is prudent to keep one reserved. As such, there is a single value for large stage candidates, and the no version attribute is the prevalent attribute for existing stage objects.